УДК 004.932.4

БЫСТРЫЕ МОРФОЛОГИЧЕСКИЕ ПРЕОБРАЗОВАНИЯ ДЛЯ ЗАДАЧ КОРРЕКЦИИ И ПРЕОБРАЗОВАНИЯ БИНАРНЫХ ИЗОБРАЖЕНИЙ

А.В. Афонасенко

Томский государственный университет
E-mail: Anuta8181@mail.ru

Представлены быстрые алгоритмы морфологических преобразований бинарных изображений. Отличительной особенностью представленных алгоритмов является исключение операций поиска минимального или максимального элемента внутри структурирующего элемента, что существенно повышает скорость реализации данных алгоритмов.

Использование методов морфологического анализа изображения для коррекции изображений, искаженных случайной некоррелированной помехой является одним из перспективных направлений в области анализа изображений [1–8]. Для оперативных систем обработки и анализа изображений важную роль играют их скоростные характеристики. Поэтому построение эффективных, быстрых алгоритмов морфологических преобразований изображения является актуальной задачей.

Пусть R и Z представляют соответственно множества действительных и целых чисел, и пусть E представляет d-мерное непрерывное пространство R^d, (d=1,2,3,...,K) или дискретное пространство Z^d. Тогда d-мерный сигнал можно представить как функцию области R^d (непрерывной) или Z^d (дискретной), диапазон которой составляет либо R – при непрерывном изменении амплитуды, либо Z – при квантованном изменении амплитуды. Двоичные сигналы могут быть представлены с помощью множеств. Например, изображение в левой верхней части рис. 1 представляет двоичный сигнал, в котором область белого фона выражена через 0, а заштрихованная зона через 1. Ясно, что сигнал может быть также представлен множеством X точек, соответствующих заштрихованной зоне.

Двоичные изображения часто получают в результате пороговой селекции полутоновых изображений, и пороговая селекция часто используется для того, чтобы представить полутоновые изображения через двоичные сигналы, то есть с помощью множеств. Серра [1] использует представление действительной d-мерной функции $f(x)$, где x означает d-мерный вектор, с помощью ансамбля его d-мерных пороговых множеств, определяемых как

$$T_a(f) = \{x : f(x) \geq a, -\infty < a < \infty\},$$

где амплитуда a полностью перекрывает R или Z, в зависимости от того, имеет ли сигнал f непрерывный или квантованный диапазон. Пороговые множества линейно упорядочены, поскольку $a<b \Rightarrow T_a(f) \supseteq T_b(f)$, и позволяют однозначно восстановить сигнал, так как

$$f(x) = \max\{a : x \in T_a(f)\}, \quad \forall x. \qquad (1)$$

Сигналы $f_a(x)$ несут ту же информацию, что и пороговые множества $T_a(x)$. Следовательно, $f(x)$ может быть восстановлено по ансамблю $f_a(x)$, поскольку

$$f(x) = \max\{a : f_a = 1\}, \quad \forall x.$$





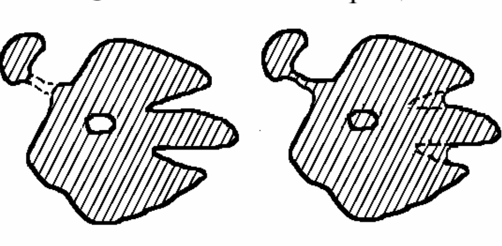


Рис. 1. *Эрозия, наращение, размыкание и замыкание множества точек X с помощью структурирующего элемента B. Штриховые области соответствуют внутренности множеств, сплошная черная линия обозначает границы преобразуемых множеств, а пунктирная линия – границу исходного множества точек X*

Фундаментальными морфологическими операторами для множеств являются наращение $\oplus$ и эрозия $\bigcirc$ с помощью B, B – структурирующий элемент, представляющий собой компактное множество малого размера и простой формы. Операция наращения всех пороговых множеств функции f с помощью одного и того же компактного множества дает множества, $T_a(f)\oplus B$, которые являются пороговыми множествами новой функции $f\oplus B$, называемой наращением функции f с помощью B. Эта новая функция может быть вычислена либо из (1) как $(f\oplus B)(x)=\max\{a{:}x\in T_a(f)\oplus B\}$, или из прямой эквивалентной формулы

$$(f \oplus B)(x) = \max_{y\in B}\{f(x-y)\}.$$

Подобно этому операция эрозии всех пороговых множеств функции f с помощью одного и того же множества B и суперпозиции всех выходных множеств посредством (1) дает новую функцию, называемую эрозией функции f с помощью множества B, которая может быть вычислена по эквивалентной формуле

$$(f \bigcirc B)(x) = \min_{y\in B}\{f(x+y)\}.$$

Другие операторы могут быть определены как комбинации эрозии и наращения. Например, два дополнительных фундаментальных оператора – размыкание $\circ$ и замыкание $\bullet f$ с помощью B определяются как

$$f \circ B = (f \bigcirc B) \oplus B,$$

$$f \bullet B = (f \oplus B) \bigcirc B.$$

Для иллюстрации геометрического поведения этих операторов полезно рассмотреть такие двумерные множества, как множество и структурирующий элемент B, показанные в верхней части рис. 1. Этот рисунок иллюстрирует, что центр структурирующего элемента располагается на границе преобразуемого множества; эрозия приводит к уменьшению множества X, а наращение – к его увеличению. Размыкание подавляет острые выступы и прорезает узкие перешейки в X, тогда как замыкание заполнят узкие заливы и малые отверстия, и таким образом $X\bigcirc B\subseteq X\subseteq X\bullet B$. Следовательно, если структурирующий элемент B имеет регулярную форму, размыкание и замыкание можно рассматривать как нелинейную фильтрацию, которая сглаживает контуры входного сигнала. Ясно, что форма и размер структурирующего элемента определяют природу и степень сглаживания.

Таким образом, алгоритм реализации морфологических операций для полутонового изображения заключается в следующем. Создается массив, такого же типа как изображение, который имеет такие же размеры как исходное изображение. В него записывается результат морфологической операции. Выбирается окно размером структурирующего элемента. Окно перемещается попиксельно по всему изображению и для точки оказавшейся в центре окна выбирается минимальное или максимальное (в зависимости от выполняемой операции, эрозия или наращение) значение из значений пикселей находящихся в пределах окна и это значение записывается в массив.

В реальных технических системах, в которых решаются такие задачи как, идентификация номерных знаков автомобилей, распознавание текста, анализ текстуры ткани и топологии печатных плат, возникает необходимость в обработке бинарных изображений. В подобных системах требуется высокая точность и скорость реализации алгоритмов при минимальных затратах памяти. Традиционные методы морфологических преобразований в данном случае являются не эффективными.

Так как бинарные изображения представлены только двумя градациями яркости, например, 0 и 1, то выбор минимального или максимального элемента, в пределах окна структурирующего элемента, для каждой точки изображения является неэффективным. В этой связи нами предлагается модификация алгоритма выполнения базовых морфологических операций, эрозии и наращения. Полагаем, что форме принадлежат точки с интенсивностью 1. Структурирующий элемент выбираем размером $b \times b$ (структурирующий элемент может быть выбран любой формы, в данном случае для упрощения рассмотрения он выбран в форме квадрата), x и y – координаты точки сканирования, Nx – ширина изображения, $i(x,y)$ – значение интенсивности в точке (x,y), Ny – высота изображения. Алгоритм имеет следующие шаги:

1. создается массив, такого же типа как изображение, размером $Nx \times Ny$, в котором будет содержаться результат операции;
2. координатам x и y присваиваем по значению $b/2$;
3. вычисляем значение $i(x,y)$;
4. если $i(x,y)=1$, тогда вычисляем по очереди значение интенсивностей следующих точек: $(x-1,y)$, $(x-1,y-1)$, $(x,y-1)$, $(x+1,y-1)$, $(x+1,y)$, $(x+1,y+1)$, $(x,y+1)$, $(x-1,y+1)$;
5. если $i(x,y)=0$, то идем на шаг 7;
6. если одно из значений интенсивностей, вычисленных на шаге 4, равняется 0, то центр окна $b \times b$ располагается в точке (x,y) и всем точкам в массиве, попадающим в это окно, если это операция эрозия присваивается значение 0, или, если это операция наращения 1;
7. точке (x,y) в массиве присваиваем значение 0;
8. если $x<Nx-b/2$ и $y<Ny-b/2$, тогда выполняем $x=x+1$ и идем на шаг 2;
9. если $x=Nx-b/2$ и $y<Ny-b/2$, тогда выполняем $x=b/2$, $y=y+1$ и идем на шаг 2;
10. конец алгоритма.

Оценка скорости реализации алгоритма показывает, что количество операций присвоения точке значения интенсивности будет пропорционально лишь количеству точек в форме умноженное на размер структурирующего элемента. Отличительной особенностью алгоритма является тот факт, что в нем исключены операции поиска минимального или максимального элемента внутри структурирующего элемента.

Далее приводится описание модификации алгоритма реализации морфологических операций размыкания и замыкания для бинарных изображений. Так, как смысл морфологически операций размыкания и замыкания заключается в том, что размыкание подавляет острые выступы и прорезает узкие перешейки в X (рис. 1), а замыкание заполняет узкие заливы и малые отверстия, то предлагается следующая модификация алгоритма выполнения операций размыкания и замыкания для бинарных изображений. Рассмотрим алгоритм реализации операции замыкания по шагам (для операции размыкания выполняется то же самое, что и для замыкания, только вместо нуля единица, а вместо единицы ноль).

Выбираем те же обозначения, что и для базовых морфологических операций, k-счетчик.

1. Координатам x и y присваиваем по значению 0, $k=0$;
2. вычисляем значение $i(x,y)$;
3. если $i(x,y)=1$, идем на шаг 9;
4. если $i(x,y)=0$, вычисляем $i(x+1,y)$;
5. если $i(x+1,y)=0$, выполняем $k=k+1$, $x=x+1$ и вычисляем значение $i(x,y)$ до тех пор, пока $i(x,y)$ не станет равным 1 или пока x не станет равным Nx;
6. если $x=Nx$, тогда $k=0$ и идем на шаг 10;
7. если $i(x,y)=1$ и $k<=b$, тогда всем точкам от $(x-k,y)$ до (x,y) присваиваем значение 1;
8. $k=0$;
9. если $x<Nx$ и $y<Ny$, $x=x+1$ и идем на шаг 2;
10. если $x=Nx$ и $y<Ny$, $x=0$, $y=y+1$ и идем на шаг 2;
11. координатам x и y присваиваем по значению 0, $k=0$;
12. вычисляем значение $i(x,y)$;
13. если $i(x,y)=0$, идем на шаг 19;
14. если $i(x,y)=1$, вычисляем $i(x,y+1)$;
15. если $i(x,y+1)=0$, выполняем $k=k+1$, $y=y+1$, и вычисляем $i(x,y)$ до тех пор, пока $i(x,y)$ не станет равным 1 или пока y не станет равным Ny;
16. если $y=Ny$, тогда $k=0$ и идем на шаг 20;
17. если $i(x,y)=1$ и $k<=b$, тогда всем точкам от $(x,y-k)$ до (x,y) присваиваем значение 1;
18. $k=0$;
19. если $y<Ny$ и $x<Nx$, $y=y+1$ и идем на шаг 12;
20. если $y=Ny$ и $x<Nx$, x=0, $x=x+1$ и идем на шаг 12;
21. конец алгоритма.

Оценка скорости реализации алгоритма показывает, что количество операций присвоения точке значения интенсивности будет пропорционально лишь количеству точек одинаковой интенсивности располагающихся в промежутке, размером равным или меньшем, чем размер структурирующего элемента, между точками другой интенсивности. Отличительной особенностью алгоритма является тот факт, что в нем исключены операции поиска мини-

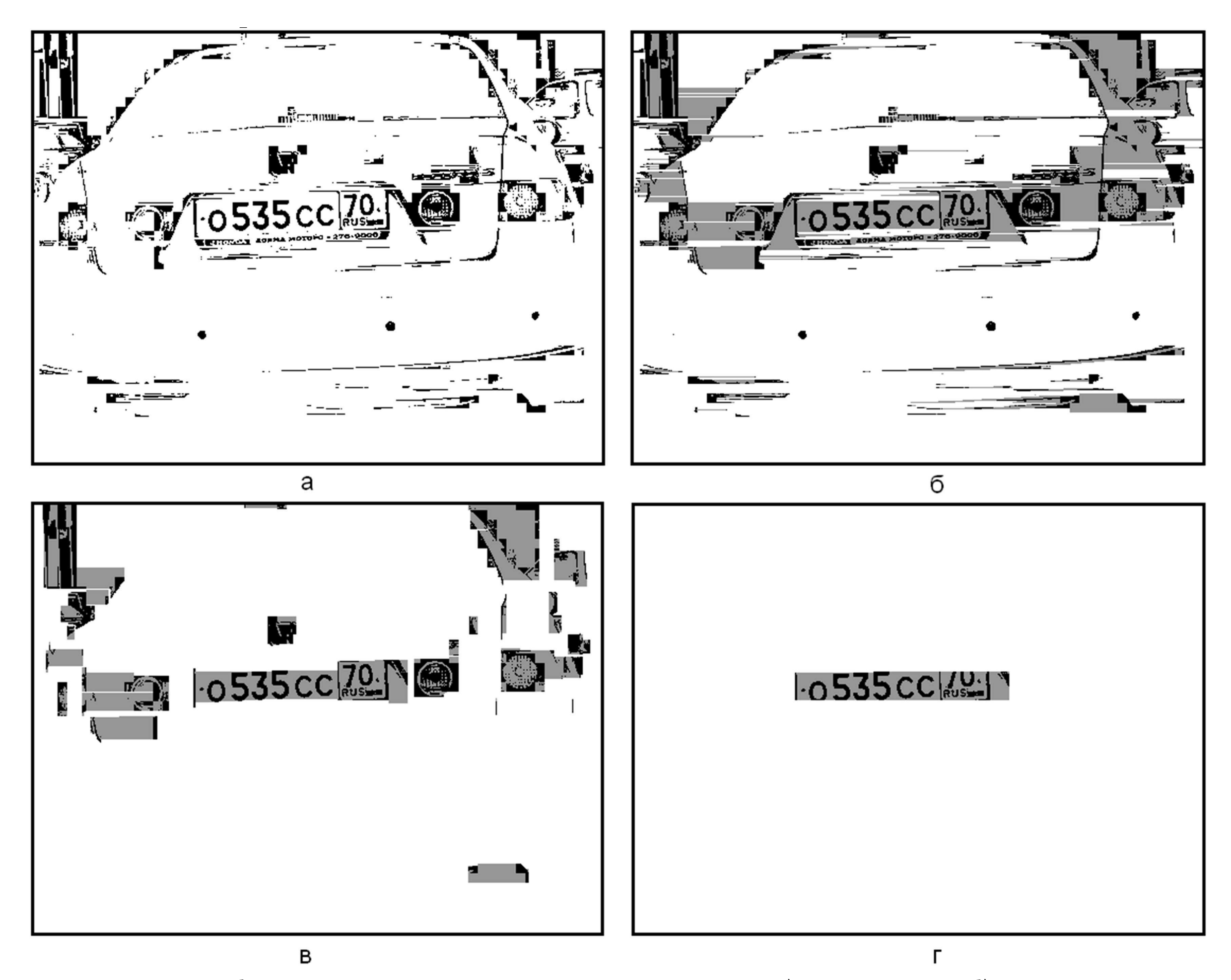


Рис. 2. *Применение быстрых алгоритмов операций замыкания и размыкания: а) исходный сигнал, б) замыкание структурирующим элементом 5×1, в, г) размыкание структурирующим элементом 1×3 и 15×1*

мального или максимального элемента внутри структурирующего элемента. А также исключена повторная замена точек форм, которая бы осуществлялась при поочередном использовании модифицированных операций эрозии и наращения.

Описанные выше быстрые алгоритмы операций замыкания и размыкания применяются для предварительной обработки изображений автомобиля, в задаче обнаружения номерной пластины. На рис. 2 представлено последовательное применение набора морфологических операций к бинаризованному изображению. На рис. 2, *б*, представлен результат применения операции замыкание структурирующим элементом размером 5×1. Размер структурирующего элемента равняется расстоянию между буквами на номерном знаке автомобиля, т. е. размеры номерной пластины должны быть заранее известны.

Результат для наглядности представлен черным и серым цветом, т. е. точки с интенсивностью 0 и 150 принадлежат одной форме.

На рис. 2, в представлен результат операции размыкания, примененной к изображению рис. 2, *б*, размер структурирующего элемента выбран размером меньшим, чем высота букв 1×3. На рис. 2, *г* – результат операции размыкания, примененной к изображению рис. 2, *в*, размер структурирующего элемента выбран размером меньшим, чем длина номерной пластины 15×1.

Данный метод, обладает следующими недостатками. Это проблема выбора порога бинаризации. Данную проблему, возможно, решить с помощью алгоритмов адаптивной бинаризации. Второй недостаток связан с трудностью определения размеров структурирующего элемента, т. к. он связан с размерами символов.

Программа была разработана и реализована в визуальной среде программирования Delphi7. Во время проведения испытаний, на компьютере с конфигурацией AMD Athlon 2,1 ГГц и оперативной памятью объемом 256 Мб, для изображения размером 1024×768 время исполнения составило 20 мс. Время реализации морфологических операций традиционным путем составило 900 мс.

Отличительной особенностью предложенных алгоритмов является исключение операции поиска минимального или максимального элемента внутри структурирующего элемента, что существенно

уменьшает время реализации данных алгоритмов для бинарных изображений. Для операций замыкания и размыкания исключена повторная замена точек форм, которая бы осуществлялась при поочередном использовании модифицированных операций эрозии и наращения.

Работа выполнена при поддержке РФФИ, проект № 06-8-00751.

СПИСОК ЛИТЕРАТУРЫ

1. Маргос П., Серра Дж. Морфологические системы для обработки многомерных сигналов // Труды института инженеров по электротехнике и радиоэлектронике. – 1990. – Т. 78. – № 4. – С. 109–132.
2. Пытьев Ю.П. Морфологический анализ изображений // Доклады АН СССР. – 1983. – Т. 269. – № 5. – С. 1061–1064.
3. Пытьев Ю.П. Задачи морфологического анализа изображений // Математические методы исследования природных ресурсов Земли из космоса / Под ред. В.Г. Золотухина. – М.: Наука, 1984. – С. 41–83.
4. Абламейко С.В., Лагуновский Д.М. Обработка изображений. – Минск: Амалфея, 2000. – 304 с.
5. Chen. T. A pseudo top-hat mathematical morphological approach to edge detection in dark regions // Pattern Recognition. – 2002. – V. 35. – № 1. – P. 199–210.
6. Sossa-Azuela J.H. Computing geometric moments using morphological erosions // Pattern Recognition. – 2001. – V. 34. – № 2. – P. 271–276.
7. Laganiere R. A morphological operator for corner detection // Pattern Recognition. – 1998. – V. 31. – № 11. – P. 1643–1652.
8. Pring Z. Document filters using morphological and geometrical features of characters // Image and Vision Computing. – 2001. – V. 19. – № 12. – P. 847–855.

УДК 004.056